SISTEMI OPERATIVI e
LABORATORIO DI SISTEMI OPERATIVI
(A.A. 12-13) – 15 GENNAIO 2014

IMPORTANTE:

1) Fare il login sui sistemi in modalità Linux usando il proprio **username** e **password**, attivare syncexam.sh e passare in modalità testuale.
2) I file prodotti devono essere collocati in un **sottodirettorio** (che deve essere nella directory studente_XXX) che deve essere creato e avere nome **ESAME15Gen14_1_01.** FARE ATTENZIONE AL NOME DEL DIRETTORIO, in particolare alle maiuscole e ai trattini indicati. Verrà penalizzata l'assenza del direttorio con il nome indicato e/o l'assenza dei file nel direttorio specificato, al momento della copia automatica del direttorio e dei file. **ALLA SCADENZA DEL TEMPO A DISPOSIZIONE VERRÀ INFATTI ATTIVATA UNA PROCEDURA AUTOMATICA DI COPIA, PER OGNI STUDENTE DEL TURNO, DEI FILE CONTENUTI NEL DIRETTORIO SPECIFICATO.**
3) Il tempo a disposizione per la prova è di **75 MINUTI** per lo svolgimento della sola parte C e di **120 MINUTI** per lo svolgimento di tutto il compito.
4) Non è ammesso **nessun tipo di scambio di informazioni** né verbale né elettronico, pena la invalidazione della verifica.
5) L'assenza di commenti significativi verrà penalizzata.
6) **AL TERMINE DELLA PROVA È INDISPENSABILE CONSEGNARE IL TESTO DEL COMPITO (ANCHE IN CASO CHE UNO STUDENTE SI RITIRI): IN CASO CONTRARIO, NON POTRÀ ESSERE EFFETTUATA LA CORREZIONE DEL COMPITO MANCANDO IL TESTO DI RIFERIMENTO.**

Esercizio

Si realizzi un programma concorrente per UNIX che deve avere una parte in Bourne Shell e una parte in C.

La parte in Shell deve prevedere **tre** parametri: il primo e il secondo devono essere nomi assoluti di direttori che identificano due gerarchie (**G1** e **G2**) all'interno del file system, mentre il terzo deve essere considerato un numero intero strettamente positivo (**K**). Il programma deve cercare (in due fasi successive) nelle gerarchie **Gi** specificate (prima **G1** e poi **G2**) tutti i direttori che contengono almeno **un** file la cui lunghezza in byte sia maggiore di **K**: si riporti il nome assoluto di tali direttori sullo standard output. **Al termine dell'intera esplorazione ricorsiva di G1 e di G2**, si deve verificare che il numero globale di file trovati in **G1** (**F0**, **F1**, ... **FN-1**) sia uguale al numero globale di file trovati in **G2** (**FF0**, **FF1**, ... **FFN-1**) e uguale a **K**: solo in tale caso, si deve invocare la parte in C passando come parametri i nomi assoluti dei file trovati **F0**, **F1**, ... **FN-1, FF0**, **FF1**, ... **FFN-1**.

La parte in C accetta un numero variabile pari **2N** di parametri maggiore o uguale a 2 (*da controllare che 2N sia pari e sia ≥ 2*) che rappresentano i nomi assoluti di file **F0**, **F1**, ... **FN-1, FF0, FF1**, ... **FFN-1** *(tutti con lunghezza maggiore di N ~~e che contengono solo caratteri ASCII~~: entrambe queste condizioni non devono essere controllate)*. Il processo padre deve generare **N processi figli** (**P0 … PN-1**) e ognuno dei processi figli deve generare un processo nipote (**PP0 … PPN-1**): i processi figli **Pj** sono associati ai file **Fj** mentre i processi nipoti **PPj** ai file **FFj (con j che, in entrambi i casi, varia da 0 a N-1)**. Ogni figlio **Pj** e ogni nipote **PPj** eseguono concorrentemente; in particolare, ognuno dei due 'tipi' di processi deve leggere, dal suo file associato, *un singolo carattere*: il processo figlio **Pj** legge il carattere *j-esimo* (**chj**) *a partire dall'**inizio** del suo file* associato **Fj**, mentre il processo nipote **PPj** legge il carattere *j-esimo* (**cchj**) *a partire dalla **fine*** del suo file associato **FFj**. Una volta letto il proprio carattere, la *coppia figlio-nipote* passa alla fase di comunicazione: in particolare, il processo nipote **PPj** comunica al processo figlio **Pj** il carattere letto (**cchj**) e il processo figlio **Pj** verifica se il carattere letto dal processo nipote PPj è **uguale** al suo carattere letto (**chj**); in tal caso, il processo figlio **Pj comunica** al padre il carattere uguale (chj==cchj), altrimenti comunica il valore **-1** *(come byte)***.** Il padre ha il compito di stampare su standard output, *rispettando l'ordine dei primi N file* (**F0**, **F1**, ... **FN-1**), l'informazione ricevuta dai figli interpretandone il significato.
Al termine, ogni processo nipote **PPj** deve ritornare al figlio **Pj** se la lettura del carattere è andata a buon fine e la stessa cosa deve fare il processo figlio **Pj** al padre. Il padre, dopo che i figli sono terminati, deve stampare, su standard output, i **PID** di ogni figlio con il corrispondente valore ritornato.